**Specification**

Please amend the Specification as shown below. Applicant submits that no new matter is being added to the Specification. The paragraph numbering reflects the numbering of paragraphs in the originally filed application.

Please replace paragraph **[0039]** with the following amended paragraph:

**[0039] Figure 3** is an illustration of objects used in connecting a repository to a VCR in one embodiment of the invention. In one embodiment, objects implementing API interface *RepositoryManager* **302** can serve as an representation of a VCR from an application program's **300** standpoint. A *Repository Manager connect()* method attempts to connect all available repositories with a current user's credentials to the VCR. By way of a non-limiting example, credentials in one embodiment can be based on Java™ Authentication and Authorization Service (available from Sun Microsystems, Inc.). Those of skill in the art will recognize that many authorization schemes are possible without departing from the scope and spirit of the present embodiment. Each available content repository **312-316** is represented by an SPI *Repository* object **306-310**. The *RepositoryManager* object invokes a *connect()* method on a set of *Repository* objects. In one embodiment, a *RepositorySession* object (not shown) can be instantiated for each content repository to which a connection is attempted. In one embodiment, the *RepositoryManager connect()* method can return an array of the *RepositorySessions* to the application program, one for each repository for which a connection was attempted. Any error in the connection procedure can be described by the *RepositorySession* object's state. In another embodiment, the *RepositoryManager connect()* method can connect to a specific repository using a current user's credentials and a given repository name. In one embodiment, the name of a repository can be a URI (uniform resource identifier).

Please replace paragraph **[0040]** with the following amended paragraph:

**[0040] Figure 4** is an exemplary content model in one embodiment of the invention. The content model is shared between the API and the SPI.

Each box in **Figure [[2]] 4** represents a class or an inheritance. Hollow tipped arrows connecting boxes indicate inheritance relationships wherein the class/interface from which the arrows emanate inherit from the class/interface to which the arrows point. Solid tipped arrows indicate that the objects of the class/interface from which the arrows emanate can contain or have references (e.g., pointers or addresses) to objects of the class/interface to which the arrows point. In one embodiment, each object in a VCR has an identifier that uniquely identifies it. An identifier can be represented by an *ID* **400** (or id). An id can contain the name of a content repository and a unique id provided to it by the repository. In one embodiment, the id class/interface can be made available through a common super class/interface **414** that can provide services such as serialization, etc.

Please replace paragraph **[0051]** with the following amended paragraph:

**[0051] Figure 8** is an illustration of a content editor in one embodiment of the invention. Content editor window **800** includes a navigation pane **802** and a context-sensitive editor window **804**. Navigation pane **802** is in "content" mode **812** such that it selectively filters out nodes that define only schemas. Content node **806** ("Laptop") has been selected. Node **806** is a child of hierarchy node "Products", which itself is a child of repository "BEA Repository". Selection of node **806** causes a corresponding content node editor to be rendered in editor window **804**. The editor displays the current values for the selected node. The content type **814** indicates that the schema for this node is named "product". In this example, the node has five properties: "Style", "Description", "Color", "SKU", and "Image". A user is allowed to change the value associated with these properties and update the VCR (via the update button **808**), or remove the node from the VCR (via the remove button **810**).

Please replace paragraph **[0052]** with the following amended paragraph:

**[0052] Figure 9** is an illustration of a schema editor in one embodiment of the invention. Schema editor window **900** includes a navigation pane

902 and a context-sensitive editor window **904**. Navigation pane **902** is in "type" mode **910** such that it only displays nodes that have schemas but no content. Schema node **906** ("product") has been selected. Node **906** is a child of repository "BEA Repository". Selection of node **906** causes a corresponding schema editor to be rendered in editor window **904**. The editor displays the current schema for the selected node (e.g., derived from *ObjectClass, PropertyDefinition, PropertyChoice* objects). In this example, the node has five property definitions: "Style", "Description", "Color", "SKU" and "Image". For each property, the editor displays an indication of whether it is the primary property, its data type, its default value, and whether it is required. A property can be removed from a schema by selecting the property's delete button **912**. A property can be added by selecting the "add property" button **908**. A property's attributes can be changed by selecting its name **914** in the editor window or the navigation pane **906** (see **Figure 10**).

Please replace paragraph **[0053]** with the following amended paragraph:

**[0053]** **Figure 10** is an illustration of a property editor in one embodiment of the invention. Property editor window **1000** includes a navigation pane **1002** and a context-sensitive editor window **1004**. The schema named "product" is being edited. Schema properties definitions are listed beneath their schema name in the navigation pane **1002**. Schema property **1008** ("color") has been selected. The editor window **1004** displays the property's current attributes. The name of the attribute (e.g., "color"), whether the attribute is required or not, whether it is read-only, whether it is the primary property, its data type, default value(s), and whether the property is single/multiple restricted/unrestricted can be modified. Changes to the [[a]] property's attributes can be saved by selecting the update button **1006**.

Please replace paragraph **[0055]** with the following amended paragraph:

**[0055]** Referring to **Figs. 1** and **11**, content mining system (CMS) **1100** can extract content from file systems and/or websites and transfer this

content (or a reference/link to the content) to one or more repositories **108** via VCR **100**. In one embodiment, the CMS can traverse a file system and/or website and identify content therein in the form of files, HTML or XML documents, images, sounds, and/or any other kind of suitable content. This content can then be provided along with the appropriate schema information to the VCR via the API **104** for inclusion in one or more repositories **108** via the SPI **102**. In one embodiment, content can be mapped to content nodes and directories can be mapped to hierarchy nodes. This can preserve the hierarchical relationship of content that arises from the organization of a file system or a website.

Please replace paragraph **[0064]** with the following amended paragraph:

**[0064]** In one embodiment, filter process can interact with the CMS via an API ~~1~~**104** or some other suitable mechanism. The API ~~1~~**104** can include services for allowing the CMS to direct a filter process to extract properties from the content (or provides references/links to the content properties). In one embodiment, a filter process can be an object accessible by the CMS. In another embodiment, a default filter process can be provided for content types that the CMS does not recognize.